Supplement to Wininger M & Muñoz CX, Models of hydration and nutrition require environmental data, Public Health Nutrition. Accepted for publication: July 26, 2019.

**SUPPLEMENT 1: CODE**

This code can be copy-pasted directly into R without alteration, and will yield a result. User-specific inputs are shown in blue for ease of case-testing.

################################################################

### wx\_retriever\_complete.R

### <author redacted for blinded peer review>

###

### -This code requires 4 data elements to be inputted by the

### user: latitude, longitude, participant identifier, and date

### of interest, and one additional operational parameter: total

### number of loops permitted on a given record of interest.

### this code will then retrieve weather station information

### from the national climate data center with a best-match to

### the weather station nearest to the specified coordinates.

### if no record exists for the specified day, this code will

### loop through nearby stations (up to the pre-specified loop

### limit) until a valid record is found. output will be a

### matrix containing the following additional data points:

### station identifiers, station elevation, distance between

### specified lat-lon and the matched station (meters), three

### weather variables (temperature, dew-point, and precipita-

### tion amount), and length-of-day information (hours between

### dawn and dusk).

###

### NOTE: In order to sustain loop operation, all web errors

### are handled similarly, by targeting the next station. This

### applies both to non-existent files, and also busy server

### errors. This, operating this code at a time of high web

### traffic may yield different results (server denial loops to

### a farther weather station), versus operating code at a time

### of low-traffic (nearest station is accessed on first

### attempt).

###

################################################################

### close all, clear all

rm(list=ls(all=TRUE))

graphics.off()

### load library

library(GSODR) # reformat\_GSOD

library(geosphere) # distm

library(suncalc) # getSunlightTimes

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### USER INPUT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### declare the latitude and longitude and dates of interest,

### set a maximum number of attempts to retrieve data from the

### web

### declare lat-lon

latlon=rbind( c(+034.0522,-118.2437), # los angeles

c(-000.1807,-078.4678), # quito

c(+008.9806,+038.7578), # addis ababa

c(+035.6762,+139.6503), # tokyo

c(+027.7172,+085.3240), # kathmandu

c(-000.1807,-078.4678), # quito

c(-025.7479,+028.2293)) # pretoria

colnames(latlon)=c("lat","lon")

rownames(latlon)=c("Pt\_001","Pt\_002","Pt\_003","Pt\_004","Pt\_005","Pt\_006","Pt\_007")

### declare dates

date=c("04/15/2015","5/5/2018","03/15/2016","7/18/2012","10/16/2014","11/30/2018","5/5/2018")

### declare a maximumum number of web retrievals

max\_retr=20

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### SUB-ROUTINES ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### function to test ncdc.gov for file access, will seamlessly

### return to calling function if URL yields error

webReader<- function(url){

tryCatch(

expr = {

webinfo=reformat\_GSOD(file\_list=url)

return(webinfo)

},

error = function(e){

return(NA)

},

warning = function(w){

return(NA)

}

)

}

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### DATA RETRIEVAL ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### retreive weather archive data from the web

### retrieve station meta-data from source (29778 x 11)

meta\_data=read.table("ftp://ftp.ncdc.noaa.gov/pub/data/noaa/isd-history.csv",sep=",",header=TRUE,fill=TRUE)

### merge user-specified data into a single data-frame

info\_frame=as.data.frame(cbind(latlon,date),check.names=FALSE)

info\_frame$date=as.Date(info\_frame$date,format="%m/%d/%Y")

info\_frame$lat=as.numeric(as.character(info\_frame$lat))

info\_frame$lon=as.numeric(as.character(info\_frame$lon))

### declare prefix for URL

url\_pr="ftp://ftp.ncdc.noaa.gov/pub/data/gsod"

### initialize vectors for storing weather data

t=NULL # temperature

d=NULL # dew point

p=NULL # precipitation

lt=NULL # latitude

ln=NULL # longitude

u=NULL # usaf

w=NULL # wban

### for each row-item of user-specified data

for (i in 1:nrow(info\_frame)){

### sort all stations in distance order

i\_latlon=cbind(info\_frame$lon[i],info\_frame$lat[i])

d\_latlon=cbind(meta\_data$LON,meta\_data$LAT)

dist\_vals=distm(i\_latlon,d\_latlon)

meta\_sort=meta\_data[order(dist\_vals,decreasing="FALSE"),]

### initialize a loop flag

loop\_flag=TRUE

### initialize a retriever tally

retr\_tally=1

### while flag permits looping

while (loop\_flag){

### construct URL for next-nearest station

year\_val=substr(info\_frame$date[i],1,4)

usaf=meta\_sort$USAF[retr\_tally]

wban=meta\_sort$WBAN[retr\_tally]

read\_url=paste(url\_pr,"/",year\_val,"/",usaf,"-",wban,"-",year\_val,".op.gz",sep="")

### attempt to retrieve data from this station-year

sttn\_data=webReader(read\_url)

### if there is no station-year data

if (is.na(sttn\_data)){

### continue looping

loop\_flag=TRUE

### else, if there is station-year data

}else{

### continue to loop if no station-day found

loop\_flag=length(which(sttn\_data$YEARMODA %in% info\_frame$date[i]))==0

}

### if no station-day found

if (loop\_flag){

### increment retriever tally

retr\_tally=retr\_tally+1

### if loop limit reached

if (retr\_tally>max\_retr){

### stop looping

loop\_flag=FALSE

### log empty values

t=c(t,NA)

d=c(d,NA)

p=c(p,NA)

lt=c(lt,NA)

ln=c(ln,NA)

u=c(u,NA)

w=c(w,NA)

}

### else, if a station-day was found

}else{

### extract weather and station information

row\_idx=which(sttn\_data$YEARMODA %in% info\_frame$date[i])

t=c(t,sttn\_data$TEMP[row\_idx])

d=c(d,sttn\_data$DEWP[row\_idx])

p=c(p,sttn\_data$PRCP[row\_idx])

lt=c(lt,sttn\_data$LAT[row\_idx])

ln=c(ln,sttn\_data$LON[row\_idx])

u=c(u,sttn\_data$USAF[row\_idx])

w=c(w,sttn\_data$WBAN[row\_idx])

### and stop looping

loop\_flag=FALSE

} #~~ end-if on valid station-day result

} #~~ end-while on loop through nearby stations

} #~~ end-for on loop through location-days of interest

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### DATA CONDITIONING & DATA FUSION ~~~~~~~~~~~~~~~~~~~~~~~~~###

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### clean data and merge datasets

### compile information onto information frame

info\_frame=cbind(info\_frame,t,d,p,lt,ln,u,w)

repl\_names=c("TEMP","DEWP","PRCP","LAT","LON","USAF","WBAN")

names(info\_frame)[(ncol(info\_frame)-6):ncol(info\_frame)]=repl\_names

### compute distance between location and station

dist\_matx=distm(cbind(info\_frame$lon,info\_frame$lat),cbind(info\_frame$LON,info\_frame$LAT))

info\_frame$DIST=diag(dist\_matx)

### add elevation information

info\_paste=paste(info\_frame$USAF,info\_frame$WBAN,sep="")

meta\_paste=paste(meta\_data$USAF,meta\_data$WBAN,sep="")

meta\_inds=match(info\_paste,meta\_paste)

info\_frame$ELEV=meta\_data$ELEV[meta\_inds]

### obtain day-length information

daylight=NULL

for (i in 1:nrow(info\_frame)){

d=date = info\_frame$date[i]

lat=info\_frame$lat[i]

lon=info\_frame$lon[i]

sunlight=getSunlightTimes(date = d, lat = lat, lon = lon)

daylight=c(daylight,as.numeric(sunlight$dusk-sunlight$dawn))

}

info\_frame$DAYL=daylight

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### OUTPUT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

###~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~###

### print matrix of final merged data

### display information

print(info\_frame)

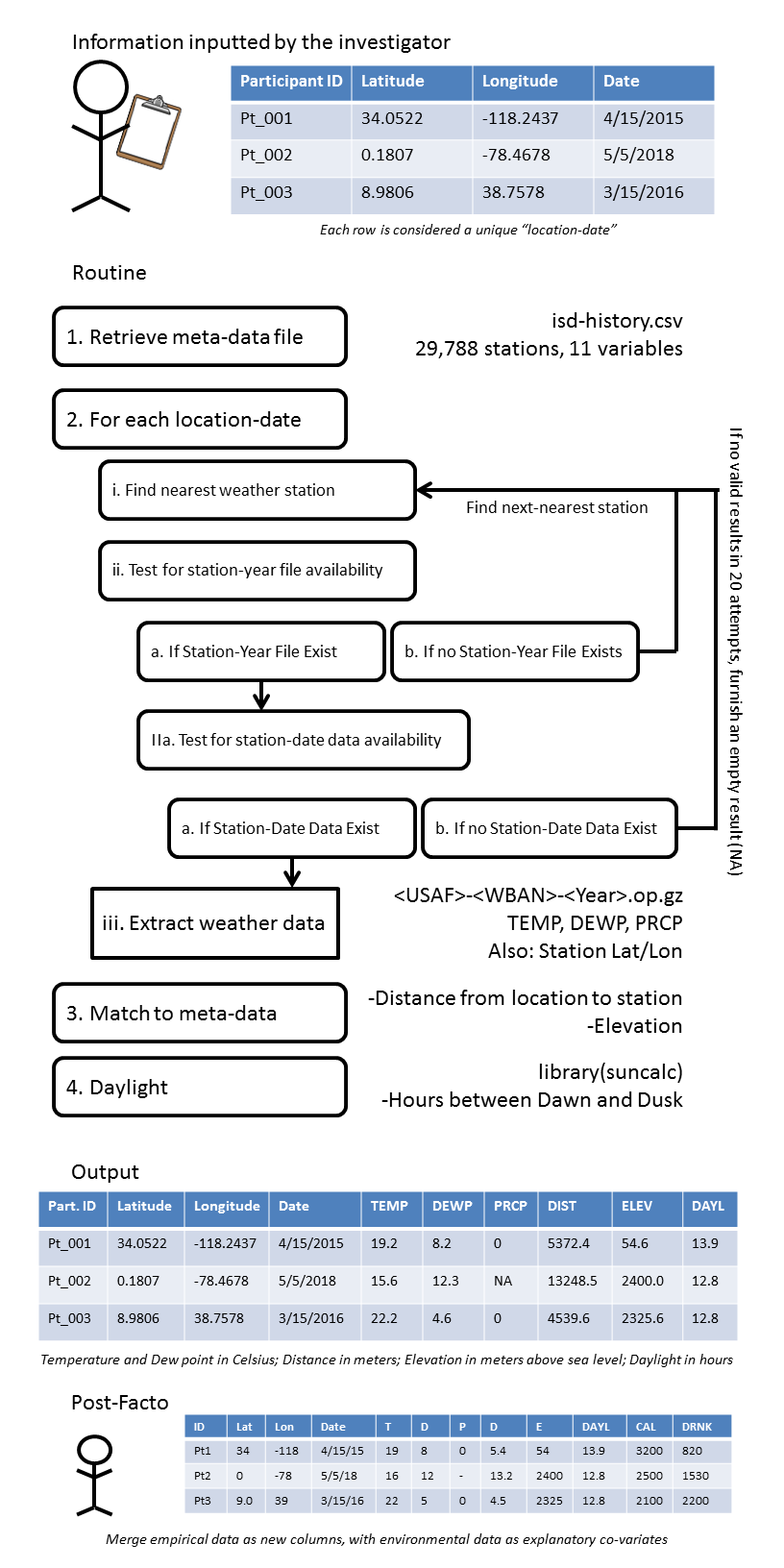
**SUPPLEMENT 2: CODE NOTES**

1. This code takes for items as input from the investigator: Participant identifier, Latitude- and Longitude of participant’s location, and Date of interest.
2. Input is currently posed as manual entry into the R-code, but can be readily reconfigured as import from a spreadsheet
3. Output is currently posed as display at the command prompt, but can be readily reconfigured as export to a spreadsheet
4. Routine will make 20 attempts to retrieve web-data for a given location-date. If no valid data are obtained after 20 attempts, empty data will be supplied. Note that this is most likely to depend on two factors: whether station data exist for the target date, and whether the NCDC servers are open for connection. Routine success is sensitive to web traffic at NCDC; it will be best to run this routine at a low-traffic time. NCDC will provide temperature, dew-point and precipitation information; NCDC meta-data will be used to find the distance between the participant’s location and the weather station used to provide the environmental data; meta-data will also be used to provide elevation information. Daylight information will be provided through the suncalc library in R, using latitude, longitude, and day information.
5. Upon completion, empirical data can be added as new columns, yielding a new dataset in long-format with environmental data as explanatory co-variates
6. Code requires a working internet connection, and assumes stability of web-resources.
7. To use dates in other formats, modify the line of code as:

info\_frame$date=as.Date(info\_frame$date,format="%m/%d/%Y")

1. Note that the provided example will yield different weather stations for Participant 2 versus Participant 6 (from identical latitude-longitude). Weather station 840720-99999 is closer (2.2km), but did not report on 5/5/18; instead, station 840725-99999 (2.4km) was accessed. This serves to demonstrate the functionality of the code in the case of contingency (missing station-day).

**SUPPLEMENT 3: CODE FLOW DIAGRAM**



**SUPPLEMENT 4: OUTPUT AND EMPIRICAL DATA MERGER**

